Stepwise General Relativity Calculations with CAS
Using SageManifolds

Tae Geun Kim
Department of Physics, Yonsei University[*]
(Dated: December 19, 2017)

General Relativity (GR) is one of main columns in modern physics. Its great geometrical structure not only provides beauty but also offers extremely accurate calculation results. But since its nonlinearity, it's too painful to calculate with hands. However, there is a great solution to overcome it - Computer Algebra System (CAS). CAS automatically calculates all the variables of the GR when you insert only the initial conditions. Although there are some good CAS programs now such as Mathematica, Maple, Maxima, Sympy, SageManifolds and etc, some of them are too expensive to students or not too rigorous to do something with Geometry. SageManifolds can be good choice to solve these problems. In this paper, I'll use SageManifolds to calculate geometric objects in General Relativity for specific case - Friedmann-Robertson-Walker Metric.

I. INTRODUCTION

A. Necessity of CAS

There are two greatest accurate calculations by human being - Quantum Electrodynamics & General Relativity (GR). In particular, GR had become a representative of elegant physics due to its structural beauty. We can represent GR with just one equation:

$$G_{\mu\nu} + \Lambda g_{\mu\nu} = 8\pi G_N T_{\mu\nu} \tag{1}$$

This short equation can explain the origin of gravity, blackhole, gravitational wave and even cosmic expansion. Like many other fields of physics, its computation procedure is mechanistic. Thus, almost all of GR computing procedures are same except input variables, for example, metric tensor. However the process from metric to Einstein equation is continuation of hardship and adversity. To get our final goal, Einstein tensor, we must go through a number of intermediate steps, including connection coefficients and Riemann tensor. In addition, each process must undergo very complex calculations. Because of this complexity, we must be helped by someone who can easily do these calculations. In the past, if such a person was a mathematician or graduate student capable of calculating, it is now a computer program. And especially in this case, Computer Algebra System (CAS) is the best choice. Now, there are a lot of great CAS such as Mathematica, Maple, Maxima, Sympy and Sagemath. There is a good reference comparing these systems [1].

B. SageManifolds

As mentioned above, there are many CASs now. Typical CASs have internal packages for general relativity calculations. For example, Mathematica has GR, Ricci packages, Maxima has ctensor package and Sagemath has SageManifolds. You can choose one of them to compute GR objects attending to your preference. I choose the SageManifolds because it allows for rigorous differential geometry calculations. Unusually, SageManifolds deals with differentiable manifolds of arbitrary dimension not just a tensor calculation. Thus in this system, a tensor field is then discribed by its representations in each vector frame. It contains lots of functionalities such as topological manifolds, exterior calculus, Lie derivatives of tensor fields, general affine connections and some plotting capabilities [2].

Due to its unique structure, SageManifolds has clear advantages and disadvantages. Although Sagemath is user friendly and convenient to use, it's not. You can't obtain any tensor calculations unless you are good at differential geometrical structure. So if this is your first time to use SageManifolds, then you will be confused by so many errors. But if you are good at differential geometry and adapt to using SageManifolds, you will be able to do graceful and effective computation.

II. PROGRESS

To obtain physical equation from General Relativity, we should need Einstein equation. In order to get the Einstein equation, it's necessary to determine geometrical structure and obtain Energy-Momentum tensor. So, let's start to determine geometrical structures with SageManifolds first.

A. Geometrical Structure

The differential geometry is based on differentiable manifolds. So, we should start with defining manifolds.

```
M = Manifold(4, 'M', '\mathcal{M}')
```

The '`Manifold`' function receive dimension, name and latex name. We want to deal with FRW metric in the

[*] edeftg@gmail.com

Minkowski spacetime, so we should input dimension 4. Then now, we should define chart (coordinate).

```
fr.<t,r,th,ph> = M.chart(r't r:[0,+oo)
    th:[0,pi]:\theta ph:[0,2*pi):\phi')
```

The '`fr.<t,r,th,ph>`' defines '`fr`'as chart which has coordinate (t, r, θ, ϕ). RHS is a little bit complex. I gave range of each coordinates through that code. Don't insert blank in range. If you do that, you can see error.

We defined manifold and chart. To play with GR, we should insert metric tensor. We use Friedmann-Robertson-Walker metric as follows.

$$g_{ab} = -dt^2 + a(t)^2\left[\frac{dr^2}{1-Kr^2} + r^2(d\theta^2 + \sin^2(\theta)d\phi^2)\right] \tag{2}$$

Ofcourse, We can use this metric tensor in SageManifolds.

```
# Declare Variables
var('Lambda K G', domain='real')
a = M.scalar_field(function('a')(t), name='a')

# Define Metric
g = M.lorentzian_metric('g')
g[0,0] = -1
g[1,1] = a^2/(1-K*r^2)
g[2,2] = a^2*r^2
g[3,3] = a^2*r^2*sin(th)^2
```

Code is simple. There were some variables except coordinates such as K, G, Λ in FRW cosmology, thus we should declare those variables first. And trivially FRW theory allows only real variables, so should check it. For metric, we usually use Lorentzian metric not Riemannian. SageManifolds has both as sub-objects in `Manifold`, so you can choose it for each theory. Since we defined 4-dimensional `Manifold`, so automatically our metric tensor has 16 components. In SageManifolds, default component value is 0, so we just input diagonal componets in this case. Putting componets is very simple. Just type.

Now, we have metric tensor. It means we can do anything in General Relativity in principle. But only with hands, just taking all connection components is too hard to do. However, it's too easy in SageManifolds.

```
nab = g.connection()
show(nab.display(only_nonredundant=true))
```

Because in SageManifolds, metric already has connection method. Just type `g.connection()`, that's it. You can see the result by 'display' method. '`only_nonredundant=true`' means just display non vanishing components by symmetry.

$$\Gamma^{t}{}_{r\,r} = -\frac{a(t)\frac{\partial a}{\partial t}}{Kr^2 - 1}$$
$$\Gamma^{t}{}_{\theta\,\theta} = r^2 a(t)\frac{\partial a}{\partial t}$$
$$\Gamma^{t}{}_{\phi\,\phi} = r^2 a(t)\sin(\theta)^2\frac{\partial a}{\partial t}$$
$$\Gamma^{r}{}_{t\,r} = \frac{\frac{\partial a}{\partial t}}{a(t)}$$
$$\Gamma^{r}{}_{r\,r} = -\frac{Kr}{Kr^2 - 1}$$
$$\Gamma^{r}{}_{\theta\,\theta} = Kr^3 - r$$
$$\Gamma^{r}{}_{\phi\,\phi} = \left(Kr^3 - r\right)\sin(\theta)^2$$
$$\Gamma^{\theta}{}_{t\,\theta} = \frac{\frac{\partial a}{\partial t}}{a(t)}$$
$$\Gamma^{\theta}{}_{r\,\theta} = \frac{1}{r}$$
$$\Gamma^{\theta}{}_{\phi\,\phi} = -\cos(\theta)\sin(\theta)$$
$$\Gamma^{\phi}{}_{t\,\phi} = \frac{\frac{\partial a}{\partial t}}{a(t)}$$
$$\Gamma^{\phi}{}_{r\,\phi} = \frac{1}{r}$$
$$\Gamma^{\phi}{}_{\theta\,\phi} = \frac{\cos(\theta)}{\sin(\theta)}$$

This is the result of above code. SageManifolds has latex auto-lendering feature, so you can get beautiful results or latex code directly. This feature is so helpful to write articles.

You can also check that this is torsion free and see really the metric is parallel transported with the following code:

```
show(nab.torsion().display())
show(nab(g).display())
```

The main object of geometric part of GR is curvature. Now, let's see how to deal with it in SageManifolds.

```
Ric = nab.ricci()
show(Ric.display())
```

In SageManifolds, connection & metric have '`ricci()`' method both. You can get Ricci tensor from just a line of code.

$$\begin{aligned}\mathrm{Ric}\left(g\right) &= -\frac{3\,\frac{\partial^2 a}{\partial t^2}}{a\left(t\right)}\mathrm{d}t\otimes\mathrm{d}t \\ &+\left(-\frac{2\,\left(\frac{\partial a}{\partial t}\right)^2 + a\left(t\right)\frac{\partial^2 a}{\partial t^2} + 2\,K}{Kr^2-1}\right)\mathrm{d}r\otimes\mathrm{d}r \\ &+\left(2\,r^2\left(\frac{\partial a}{\partial t}\right)^2 + r^2 a\left(t\right)\frac{\partial^2 a}{\partial t^2} + 2\,Kr^2\right)\mathrm{d}\theta\otimes\mathrm{d}\theta \\ &+\left(2\,r^2\left(\frac{\partial a}{\partial t}\right)^2 + r^2 a\left(t\right)\frac{\partial^2 a}{\partial t^2} + 2\,Kr^2\right)\sin\left(\theta\right)^2 \\ &\quad \mathrm{d}\phi\otimes\mathrm{d}\phi \end{aligned} \tag{3}$$

Except internal method, SageManifolds provides index product (contraction or tensor product). So, we can use it to obtain Ricci scalar.

```
R = g.inverse()['^ab']*Ric['_ab']
```

'`<Tensor>['_ab']`' or '`<Tensor>['^ab']`' means T_{ab} or T^{ab}. And we can implement either contraction or tensor products simply by multiplying them.

Finally, we can get Einstein tensor.

```
Gab = Ric - 1/2*R*g
Gab.set_name(r'G_{ab}')
show(Gab.display())
```

$$\begin{aligned}G_{ab} &= \frac{3\left(\left(\frac{\partial a}{\partial t}\right)^2 + K\right)}{a\left(t\right)^2}\mathrm{d}t\otimes\mathrm{d}t \\ &+\left(\frac{\left(\frac{\partial a}{\partial t}\right)^2 + 2\,a\left(t\right)\frac{\partial^2 a}{\partial t^2} + K}{Kr^2-1}\right)\mathrm{d}r\otimes\mathrm{d}r \\ &+\left(-r^2\left(\frac{\partial a}{\partial t}\right)^2 - 2\,r^2 a\left(t\right)\frac{\partial^2 a}{\partial t^2} - Kr^2\right)\mathrm{d}\theta\otimes\mathrm{d}\theta \\ &-\left(r^2\left(\frac{\partial a}{\partial t}\right)^2 + 2\,r^2 a\left(t\right)\frac{\partial^2 a}{\partial t^2} + Kr^2\right)\sin\left(\theta\right)^2 \\ &\quad \mathrm{d}\phi\otimes\mathrm{d}\phi \end{aligned} \tag{4}$$

B. Matter Information

According to GR, to generate gravity, we should need matter which is source of gravity. Inserting matter information consists of the following steps.

i) Write matter Lagrangian. (by hand)

ii) Obtain Energy-Momentum tensor with Noether's thm.

iii) Set variables except coordinate variables.

iv) Insert Energy-Momentum tensor with variables.

In case of FRW cosmology, there are well defined Energy-Momentum tensor as known as perfect fluid.

$$T_{ab} = (\rho + P)u_a u_b + P g_{ab} \tag{5}$$

So, let's start from third step.

```
# Declare scala fields
rho = M.scalar_field(function('rho')(t),
      name='rho', latex_name=r'\rho')
P = M.scalar_field(function('P')(t), name='P')

# Declare Vector Fields
u = M.vector_field('u', latex_name=r'u^a')
u[0] = 1

# Vector to Covector (1-form)
u_form = u.down(g)
u_form.set_name(r'u_a')

# Define energy momentum tensor
T = (rho + P)*u_form*u_form + P*g
T.set_name(r'T_{ab}')
show(T[:])
```

SageManifolds also allows matrix representation of tensor. '`show(T[:])`' shows all components of tensor T as matrix.

$$\begin{pmatrix} \rho\left(t\right) & 0 & 0 & 0 \\ 0 & -\frac{P(t)a(t)^2}{Kr^2-1} & 0 & 0 \\ 0 & 0 & r^2 P\left(t\right) a\left(t\right)^2 & 0 \\ 0 & 0 & 0 & r^2 P\left(t\right) a\left(t\right)^2 \sin\left(\theta\right)^2 \end{pmatrix} \tag{6}$$

And we can raise or lower indices with metric or inverse metric tensor.

```
Tmix = g.inverse()['^ab']*T['_bc']
Tmix.set_name(r'{T^a}_b')
show(Tmix.display())
show(Tmix[:])
```

$${T^a}_b = \begin{pmatrix} -\rho\left(t\right) & 0 & 0 & 0 \\ 0 & P\left(t\right) & 0 & 0 \\ 0 & 0 & P\left(t\right) & 0 \\ 0 & 0 & 0 & P\left(t\right) \end{pmatrix} \tag{7}$$

We know Energy momentum tensor is conserved.

$$\nabla_a {T^a}_b = 0 \tag{8}$$

Let's obtain this in SageManifolds.

```
# Covariant Derivative of T
DTmix = nab(Tmix)
DTmix.set_name(r'\nabla T')

# Contract = Energy-Momentum Conserved
CT = DTmix['^a_ba']
print("Energy-Momentum Conserved:\n")
show(-CT[0].expr().expand() == 0)
```

$$\frac{3\,P\left(t\right)\frac{\partial}{\partial t}a\left(t\right)}{a\left(t\right)}+\frac{3\,\rho\left(t\right)\frac{\partial}{\partial t}a\left(t\right)}{a\left(t\right)}+\frac{\partial}{\partial t}\rho\left(t\right)=0 \tag{9}$$

C. Construct Einstein Equation

Now, we have Einstein tensor & Energy Momentum tensor. So, it's easy to construct Einstein equation. Typical Einstein Equation is given as:

$$G_{ab}+\Lambda g_{ab}=8\pi G T_{ab} \tag{10}$$

In SageManifolds, just type next code, we can get complete form of Einstein equation.

```
EE = Gab + Lambda*g - 8*pi*G*T
show(EE[0,0].expr().expand() == 0)
```

$$-8\,\pi G\rho\left(t\right)-\Lambda+\frac{3\,\frac{\partial}{\partial t}a\left(t\right)^{2}}{a\left(t\right)^{2}}+\frac{3\,K}{a\left(t\right)^{2}}=0 \tag{11}$$

This equation is called Friedmann equation which is main equation in FRW cosmology.

III. CONCLUSION

So far, we have looked at the calculation process of general relativity theory with SageManifolds. Maybe it's too basic example for someone, but it's enough to show effectiveness of SageManifolds. Sagemath which includes SageManifolds is the largest open source project for Computer Algebra System in the world. It contains sympy and uses python syntax. Unlike some programs like Mathematica, Sage project not only enables symbolic programming but also implements mathematical structures such as geometry, algebra, and topology on a computer. If you are engaged in academics related to mathematics, Sagemath will surely be an excellent choice.

Especially in physics, Sagemath has a tremendous advantage, the ability to get analytic expression for most well-know physics without additional cost. For general relativity, we can obtain the differential equation by simply substituting the metric tensor. Then you can solve that equation numerically by your prefer programs. This program, called Sagemath, not only gives us another tool to validate calculations, but also naturally leads to rigidity and helps to reduce research time effectively.

IV. TODO

My original purpose was to implement a mathematical structure of general relativity using a functional programming language like Haskell or Scala, but I could not implement it because of the short time. Therefore, after the end of the semester, I would like to create the framework that automatically computes the connection coefficient and Einstein equation by simply inputting the metric tensor and then uses it to perform numerical computations. This work is expected to be of great significance since it will surely be useful for future research.

[1] T. Birkandan, C. Güzelgün, E. Sirin, and M. C. Uslu, CoRR (2017), arXiv:1703.09738 [gr-qc].

[2] E. Gourgoulhon, M. Bejger, and M. Mancini, CoRR (2014), arXiv:1412.4765 [gr-qc].